CS101 Class Notes
April 25-25, 2007-05-24
Taygun Murat Yilmaz
CHAPTER 6; Static Variables, Class Dependency, Class Aggregation, Interfaces

Relationships among classes:

    Inheritance
    Aggregation
    Dependency

Static members ;

    Dependency
    Aggregation
    Interface

Static Members:

```
public class Slogan
{
        private String phrase;
        private static int count:0;

        public  Slogan(String str)
        {
                phrase str;
                count ++;
        }

        public String toString()
        {
                return(phrase);
        }

        public static int getCount()
        {
                return(count);
        }

public class SloganCounter
{
        public static void main (String[]args)
        {
                Slogan mySlogan;

                MySlogan=new Slogan("Amaç iyi yaşamaksa");
                System.out.println(MySlogan);

                MySlogan=new Slogan("Kartallar yüksekten uçar");
                System.out.println(MySlogan);

//Now print the number of Slogan objects created
                System.out.println(“NoOfSlogans created : ”+MySlogan.getCount())

                Slogan yourSlogan=new Slogan(“Hayatın tadını çıkar”);
                System.out.println(“No of Slogan” + Slogan.getcount)
```

Static : we don’t need to create objects in order to use static methods
!!!Do not need to use static in main

Dependency:
Example;

Class Dependency: In the implementation of the methods of a class we may use other objects (of the same class or oter classes)

RATIONAL NUMBER

```
public class RationalNumber
{
    private int numarator, denominator;
    public  RationalNumber(int number; int denom)
    {
        İf(denom==0)
        {
            denom=1;
        }
//make the denominator “store” the sign.
        İf(denom<0)
        {
            numer=-numer;
            denom=-denom;
        }
        numerator=numer;
        denominator=denom;
        reduce();
    }
    public RationalNumber reciprocal()
    {
        RationalNumber rn;
        Rn= ew RationalNumber(denominator,  numerator);
        Return(rn);
    }
    “RationalNumber n1= new RationalNumber(15,20);
    RatinalNumber  n2=n1.reciprocal();” (example)

    public RationalNumber copy()
```

```
        {
                Return ( new RationalNumber(numerator, denominator);
        }
        public RationalNumber add(RationalNumber op2)
        {
                İnt  commonDenom=this.denominator*op2.denominator;
                İnt num1= numerator*op2;
        .
        .
        .
        //subtract looks like add method
        //consider multipl method
        public RationalNumber multiply(RationalNumber op2)
        {
                İnt numer= numerator*op2.getNumerator();
                İnt denom=denominator*op2.denominator;
                return( new RationalNumber(numer, denom));
        }
        public RationalNumber divide(RationalNumber op2)
        {
                return multiply(op2.reciprocal());
        }
                For getting : İf (Rn1.equals(Rn2))
                                  System.out.println("R1=R2");
        public boolean equals(Rational op2)
        {
                if(tis.numerator==op2.getNumerator()  &&
this.denominator==op2.getDenominator)
                        Return(true);
                else
                        Return(false);
        }
```

CLASS AGGREGATİON:

```
        public class Student
        {
                private String firstName, lastsName;
                private String  Address        homeAddress, schoolAddress;
                public Students(String firstName, lastName,  Address home, school)
                {
                        this.firstName=firstName;
                        this.lastName=lastName;
                        home Address= home;
                        student Address= school;
                }
        }
        public class Address
        {
```

```
        private String streetAddress, city, state;
        private long zipCode;
        public Address( String streetAddress, city, state, zipCode)
        {
            this.streetAddress= streetAddress;
            this.city=city;
            this.state=state;
            this.zipCode=zipCode;
        }
    public class StudentBody
    {
        public  static void main(String [] args)
        {
        Address columbiaUniAddress;
        }
    }
```

INTERFACES:

```
public interface Complexity
{
    public void setComplexity(int complexity);
    public int getComplexity();

    public class Question implements Complexity
    {
        private String question, answer;
        private Question( String query, String result)
        {
            question=query;
            answer=result;
            CcmplexityLevel=1;
        }
    public void setComplexity( int level)
    {
        complexityLevel level;
    }
    public int getComplexity()
    {
        return (complexityLevel);
    }
    public boolean answerCorrect(String candidateAnswer)
    {
        return( answer.equalsIgnoreCase(candidateAnswer);
    }
}
```

DESIGN AN INTERFACE:

```
public interface Shape
{
        public float getArea();
        public float getCİrcumference();
}
public class Circle implements Shape
{
}

public class RationalNumber implements Comparable
{
        public int compareTo(RationlaNumber2)
        {
                return(r.numerator);
        }
}
```

QUESTIONS;

1)What is the difference between a static variable and an instance variable?

2)Describe a dependency relationship between two classes?

3)What is an aggregate object?

ANSWERS;

1)Memory space for an instance variable is created for each object that is instantiated from a class. A static variale is shared among all objects of a class.

2)A dependency relationship between two classes occurs when the one class relies on the functionality of the other. It is often referred to as a "uses" relationship

3)It is an object that has other object as instance data. That is, an aggregate object is one tat is made up of other objects